FIG. 3B shows an initial state of the $V(8,3,9)$ network in which the connections $I_1 - I_7$ of Table 1 are previously set up. [For the sake of simplicity, FIG. 3B, FIG. 3C, and FIG. 3D do not show in the diagrams the first internal links and second internal links connected to the middle switches MS7 and MS8.] The connections I_1, I_2, I_3, I_4, I_5, I_6, and I_7 pass through the middle switches MS1, MS2, MS3, MS4, MS5, MS6, and MS7 respectively. Each of these connections is fanning out only once in the input switch and fanning out three times in each middle switch. Connection I_1 from input switch IS1 fans out once into middle switch MS1, and from middle switch MS1 thrice into output switches OS1, OS2, and OS3. Connection I_2 from input switch IS1 fans out once into middle switch MS2, and from middle switch MS2 thrice into output switches OS4, OS5, and OS6. Connection I_3 from input switch IS1 fans out once into middle switch MS3, and from middle switch MS3 thrice into output switches OS7, OS8, and OS9. Connection I_4 from input switch IS2 fans out once into middle switch MS4, and from middle switch MS4 thrice into output switches OS1, OS4, and OS7. Connection I_5 from input switch IS2 fans out once into middle switch MS5, and from middle switch MS5 thrice into output switches OS2, OS5, and OS8. Connection I_6 from input switch IS2 fans out once into middle switch MS6, and from middle switch MS6 thrice into output switches OS3, OS6, and OS9. Connection I_7 from input switch IS3 fans out once into middle switch MS7, and from middle switch MS7 thrice into output switches OS1, OS5, and OS9.

Method 140 of FIG. 3A next sets up a connection I_8 from input switch IS3 to output switches OS2, OS6 and OS7 as follows. FIG. 3C shows the state of the network of FIG. 3B after the connection I_8 of Table 1 is set up. In act 142A the scheduling method of FIG. 3A finds that only the middle switch MS8 is available to set up the connection I_8 (because all other middle switches MS1-MS7 have unavailable second internal links to at least one destination switch), and sets up the connection in act 142C through switch MS8. Therefore, Connection I_8 from input switch IS3 fans out only once into middle switch MS8, and from middle switch MS8 three times into output switches OS2, OS6, and OS7 to be connected to all the destinations.

Method 140 next sets up a connection I_9 from input switch IS3 to output switches OS3, OS4 and OS8 as follows. FIG. 3D shows the state of the network of FIG. 3C after

the connection I_9 of Table 1 is set up. The scheduling method of FIG. 3A could not find a single middle switch that has links to all required destinations available to set up the connection. However in act 142B, it finds two middle switches MS1 and MS2 to together have links to all required destinations available for the connection and accordingly the connection I_9 is set up in act 142C. And so connection I_9 fans out twice in the first switch IS3 into the middle switches MS1 and MS2. Also in the middle switch MS1 it fans out twice into output switches OS4 and OS8, and in the middle switch MS2 it fans out once into output switch OS3 to be connected to all the required destinations.

Act 142 of FIG. 3A is implemented in one embodiment by acts 242A-242D illustrated in FIG. 4A. Specifically, in this embodiment, act 142A is implemented by acts 242A, 242C, and 242D wherein a loop is formed to check if a middle switch has an available link to the input switch, and also has available links to all the required destination switches. In this implementation, the same loop is also used with an additional act 242B to implement act 142B of FIG. 3A. Use of the same loop as illustrated in FIG. 4A provides efficiency by eliminating repetition of the same acts, namely acts 242A, 242C, and 242D that would otherwise have been repeated if act 142B is performed independent of act 142A (FIG. 3A). In act 242B, the method of FIG. 4A checks if another middle switch has available links to destinations that could not be reached by use of the middle switch in act 242A (described above). As illustrated in FIG. 4B, act 242B is reached when the decision in act 242A is "NO". In one specific example, acts 242A-242B of FIG. 4C are implemented by use of the information developed in act 242A, for an efficient implementation as discussed next.

FIG. 4B is a low-level flowchart of one variant of act 142 of FIG. 4A. The control to act 142 comes from act 141 after a connection request is received. In act 142A1, an index variable i is set to a first middle switch 1 among the group of middle switches that form stage 130 (FIG. 2B) to initialize an outer loop (formed of acts of 142A2, 142A3, 242B, 242C and 242D) of a doubly nested loop. Act 142A2 checks if the input switch of the connection has an available link to the middle switch i. If not control goes to act 242C. Else if there is an available link to middle switch i, the control goes to act 142A3. Act 142A3 checks if middle switch i has available links to all the destination switches of the multicast connection request. If so the control goes to act 142C1 and the

connection is set up through middle switch i. And all the used links from middle switch i to destination output switches are marked as unavailable for future requests. Also the method returns "SUCCESS". Act 242C checks if middle switch i is the last middle switch, but act 242C never results in "yes" which means it always finds at most two middle switches to set up the connection. If act 242C results in "no", the control goes to act 242D from act 242C where i is set to the next middle switch. And the outer loops next iteration starts.

If act 142A3 results in "no" the control goes to act 142B. In act 142B1 another index variable j is set to middle switch 1 to initialize an inner loop (formed of acts 142B2, 142B3, 142B4 and 142B5) of the doubly nested loop. Then the control goes to act 142B2, where the method 140 checks if middle switch j is equal to middle switch i. If middle switch j is equal to middle switch i, the control goes to act 142B4. Else if middle switch j is not equal to middle switch i, the control goes to act 142B3 where the method 140 checks if for all the destinations that have unavailable links from middle switch i have available links from middle switch j. If act 142B3 results in "yes", the connection is set up through middle switch i and middle switch j, in act 142C2. Also all the links used in act 142C2 from middle switch i and middle switch j to destination output switches for setting up the connection are marked as unavailable for future requests and the method returns "SUCCESS". If act 142B3 results in "no", the control goes to act 142B4. In act 142B4, the method 140 checks if middle switch j is last middle switch, and if so the control goes to act 142A4, if not the control goes to act 142B5 where middle switch j is set to the next middle switch. From act 142B5 the control transfers to act 142B2. And thus acts 142B2, 142B3, 142B4 and 142B5 form the inner loop stepping through all the middle switches until two middle switches are found to set up the connection. In a three-stage network of FIG. 2B with n_1 inlet links for each of r_1 input switches, n_2 outlet links for each of r_2 output switches, no more than $2*n_1+n_2-1$ middle stage switches are necessary for the network to be strictly nonblocking and hence no more than $2*n_1+n_2-1$ middle stage switches are necessary for the method of FIG. 4A to always find one or two middle switches to set up the connection.

FIG. 4C illustrates, in a flowchart, a computer implementation of one example of the scheduling method of FIG. 4B. The flowchart FIG. 4C is similar to the flowchart of